24

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/534,956	03/24/2000	Jin-sheng Shyr	032001-033	1950

7590 02/13/2004

Edwin H. Taylor
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN
12400 Wilshire Boulevard, Seventh Floor
Los Angeles, CA 90025

EXAMINER	
KENDALL, CHUCK O	
ART UNIT	**PAPER NUMBER**
2122	H

DATE MAILED: 02/13/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 09/534,956	Applicant(s) SHYR, JIN-SHENG	
	Examiner Chuck O Kendall	Art Unit 2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *24 March 2000*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. §§ 119 and 120

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

13)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application) since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

a) ☐ The translation of the foreign language provisional application has been received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121 since a specific reference was included in the first sentence of the specification or in an Application Data Sheet. 37 CFR 1.78.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .

4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: .

DETAILED ACTION

1. This action is in response to the application filed 03/24/00.

2. Claims 1 - 20 have been examined.

Specification objection

3. The abstract should be in narrative form and generally limited to a single paragraph on a separate sheet within the range of 50 to 150 words. It is important that the abstract not exceed 150 words in length since the space provided for the abstract on the computer tape used by the printer is limited. The form and legal phraseology often used in patent claims, such as "means" and "said," should be avoided. The abstract should describe the disclosure sufficiently to assist readers in deciding whether there is a need for consulting the full patent text for details.

The language should be clear and concise and should not repeat information given in the title. It should avoid using phrases which can be implied, such as, "The disclosure concerns," "The disclosure defined by this invention," "The disclosure describes," etc.

Applicants abstract exceeds the allowed length. Correction is required.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

5. Claims 1 & 3 – 20 are rejected under 35 U.S.C. 102(b) as being anticipated by Wood USPN 5,881,311.

Regarding claim 1, Wood anticipates a method of scheduling function calls in a software program in a dynamically reconfigurable computing system which includes an embedded processor and a finite number of reconfigurable, logic partitions which are each programmed by a set of configuration bits dynamically loaded into the system's configuration memory, the method comprising the steps of:

a) processing each function call identified within the software program into both a hard implementation for hard execution in said reconfigurable logic partitions wherein a set of configuration bits associated with said each function call is generated and into a soft implementation for soft execution in said embedded processor (FIG.11, part # 1103, see Get Virtual device for function call, also see 1107 for changed bit).

b) assigning each set of said configuration bits generated during said hard implementation to a virtual partition (FIG.8, part # 811, see change bit, also see 822, for virtual device partition);

c) constructing a call history model including statistical data characterizing the patterns (Col.13:53-57, see track and statistics for call history and statistical data) in calling sequence of each function call in said software program obtained from benchmark data and initially using the call history model upon invocation of said software program (Col.13:45 – 65);

d) employing a hierarchy of storage devices to form a pyramid of staging slots (FIG.3A, part# 301, see HIERARCHICAL STORAGE MANAGEMENT) for the purpose of storing and moving each virtual partition from its initial memory location in said hierarchy of storage devices having the longest associated access latency to said configuration memory on a "time-of-need' basis (also see FIG. 3B, part # 310 - 316);

e) constructing a virtual partition table of statistical data to track and allocate said staging slots for the purpose of staging said virtual partitions within said pyramid,(FIG.8, see VIRTUAL DEVICE #1) wherein prior to execution of said software program all virtual partitions are assigned to staging slots having said longest associated access

latency, and wherein, while executing said software program, virtual partitions are moved to staging slots within said pyramid having either shorter or longer access latencies on said "time-of-need" basis dependent on said statistical data within said virtual partition table(FIG.8, see PARTITION #1, and part # 812 for BACKUP BIT).

f) upon the invocation of a current function call while executing the software program (FIG.11 step 1103), activating hard implementation of said current function call if its associated virtual partition is located within said pyramid in a position ready for activation, otherwise activating soft implementation of said function call (refer to FIG.8 for pyramid,(layers) as interpreted).

g) during activation of the current function call, utilizing said call history model to determine probable next function calls to follow the current function call (FIG.11, #114, also see update cache prior to more data (next function) as interpreted);

h) during activation of the current function call, simultaneously initiating dynamic scheduling tasks to facilitate said staging of said virtual partitions through the hierarchy of storage devices dependent on said patterns in call sequence indicated in said call history model (Col.19:40-45, see scheduled event);

i) upon the completion of activation of the current function call, dynamically updating said statistical data associated with the current function call in said call history model using statistical data obtained during the execution of the current function call (FIG. 11, part # 1103, see updating cache information);

j) repeating steps (f) to (i) for said each function call until termination of execution of said software program (FIG. 11, 116).

Regarding claim 3, the method of scheduling function calls as described in Claim 1, further comprising the step of including within said virtual partition table a plurality of entries corresponding to each function call and associated virtual partition, each entry including:

a locator pointer to a linked list of address words each for locating said associated virtual partition within said pyramid (Col.10: 43 – 55);

a tenure value entry showing a desired rank for said associated virtual partitions (Col.12:10 – 20, see perspective to data storage hierarchy and entries);

call-id entry for linking back to said associated virtual Partition's function call (Col.12:23 – 25, for call-id, see "virtual device number 813 is a pointer");

in-demand entry for tracking anticipated demand for said virtua partition (Col.12:10 – 25, see segment table, Examiner anticipates table to read on in-demand entry for tracking);

a time window entry providing the upper and lower bounds for said "time-of-need";

a time-to-enter entry providing the earliest time of activation of said virtual partition (Col.12:15 – 25, for see indication of last time of backup);

a time-to-leave entry providing the latest time of deactivation of said virtual partition (Col.12:40 – 45, see time from invalid to being written and vice versa);

a prediction entry which sums up a composite probability of being activated within said upper and lower bounds of said time window (Col. 16:10 - 23, see low water mark and high water mark); and

a opportunity entry which sums up a composite payback value that can be anticipated from executing said virtual partition in said hard execution (Col.16: 5 – 10).

Regarding claim 4, the method of scheduling function calls as described in Claim 3, further including the step of storing along with each of said address words:

a rank and slot entry indicating the location of said virtual partition within a given storage device of said pyramid, and access control flags defining memory access privileges of said location of said virtual partition within said hierarchy of storage devices (Col.23:30- 45, see priority, and array).

Regarding claim 5, the method of scheduling function calls as described in Claim 1, wherein said step of constructing said statistical call history model further comprises creating a function call table including a plurality of entries corresponding to each, function call, each function call entry including:

a pointer to a linked list of probable next-calls entries (Col.12: 60 – 13:5);

a speed-up factor corresponding to a performance gain factor of said each function call executed in hard execution (Col. 22: 40 – 45, see increases);

a hard-duration time corresponding to the length of time to execute said each function call in hard execution (Col.13: 53 – 63);

a macro-set pointer which points to a linked list of virtual partition identifiers associated with said function call (Col.13:17 – 19).

Regarding claim 6, the method of scheduling function calls as described in Claim 5, further comprising the step of including within each probable next-call entry:

a probable next-call id (Col.20: 13 –15);

a list pointer to a next probable next-call in said list of probable next-calls(Col.20: 13 –15);

a probability factor of said probable next-call (14:28 – 33);

a time-gap corresponding to the separation in time between two calls in succession (Col.13: 53 – 63, see how often access had to wait for a virtual device to be free).

Regarding claim 7, the method as described in Claim 1 wherein said step of initiating dynamic scheduling tasks includes the step of performing a demand look ahead task comprising the steps of:

recursively traversing next-calls lists including a list of said probable next ,function calls in said call history model a predetermined number of (k) times to establish a tree of next-calls that is to follow said current function call, wherein k is defined as a look-ahead depth (Col.14: 7 – 13, see array);

and predicting said "time-of-need", a probability factor, and an expected payback factor, for each of said next-calls in said tree (Col.14:23 – 30, for probability see statistics and see Col.23: 37 – 40, for prioritizing based on need).

Regarding claim 8, the method as described in Claim 7, wherein, upon completion of said demand look-ahead task, said step of initialing dynamic scheduling tasks further including the step of:

prioritizing said virtual partitions associated with each of said next-calls in said tree into three orderings including a temporal order based on said "time-of-need', a

probabilistic order based on said probability, factor and an opportunistic order based on said expected payback factor (Col.23: 37 – 40, prioritizing based on need)

Regarding claim 9, the method as described in Claim 8 wherein, upon completion of said prioritizing said virtual partitions, said step of initiating dynamic scheduling tasks further including the step of performing a tenure management task comprising the steps of:

determining, a tenure value of said each of said virtual partition associated with, function calls included in said next-calls tree, said tenure value corresponding to a desired staging slot position of said each virtual partition within said pyramid, said desired staging slot position being dependent on a 'Just- in-time" principle based on said associated latency of said staging slot position (Col. 22:37 – 942, see layer allocation algorithm);

wherein said tenure management task establishes a tenure value that is 'just- in-time" with respect to said "time-of-need' (Col.22:45 - 50).

Regarding claim 10, the method as described in Claim 9 wherein, upon completion of said tenure management task said step of initiating dynamic scheduling tasks further including the step of performing a stage de-queuing task comprising the steps of:

freeing up sufficient ones of said staging slots within said pyramid to accommodate a number of slots needed as the result of a change in tenure value determined during said step of performing said tenure management task (Col.18:46 – 55, see methods for freeing data storage space).

Regarding claim 11, the method as described in Claim 10 wherein, upon completion of said stage de-queuing task, said step of initiating dynamic scheduling tasks further including the step of performing a stage en-queuing task comprising the steps of:

allocating free staging slots within said pyramid to said each virtual partition when its rank is lower than its tenure value; and moving by copying said virtual partitions into said free staging slots (Col.19:11 – 15, see "written into free segment").

Regarding claim 12, the method of scheduling function calls as described in Claim 1, wherein said hierarchy of storage devices include hard disk, system main memory, dedicated external SRAM, dedicated on-chip buffer memory, and on-chip Configuration Cache (Col.15: 50 – 55, also see FIG.4A, part # 406).

Regarding claim 13, the method as described in Claim 1, further comprising the step of storing said each set of configuration bits assigned to each virtual partition into more than one staging slot, and chaining together said more than one staging slots with address words (Col.20: 1 - 15, see changed bits and incremental backup).

Regarding claim 14, the method as described in Claim 13 further comprising the step of managing the storing of virtual partitions within said staging slots using flag fields associated with said address words (Col. 10: 19 – 30).

Regarding claim 15, the method as described in Claim 14 wherein said flag fields include:

a valid field for indicating the validity of said set of configuration bits copied into said staging slot (Col.11: 38 – 45, see array, and validate);

a lock field for prohibiting freeing-up of said staging slot (FIG.20, 2002, see lock layer);

a park field for indicating a given level within said pyramid in which said method of scheduling no longer controls movement of virtual partitions within said pyramid of staging slots and instead said movement is controlled by another computing system control mechanism (Col.16: 38 – 47); and

a persistent field for indicating said staging slot should never be reassigned a new virtual partition (FIG.20, 2002, see lock layer).

Regarding claim 16, the method as described in Claim 7, wherein said demand look-ahead task further comprises the step of determining a composite probability factor for a sequence of more than one probable next-call (Col. 12:55 – 65).

Regarding claim 17, the method as described in Claim 11, further comprising the steps of:

incremental tasking of said dynamic scheduling tasks such that said dynamic scheduling tasks are recursively performed on a rank-by-rank basis words (Col.20: 1 -

15, see changed bits and incremental backup, also see Col. 22: 40 – 45, see increases);

interrupting said incremental tasking any time a new function call is activated wherein the most critical iterations of said dynamic scheduling tasks are accomplished for scheduling of said next function call (Col.19:40 – 47, see timer interrupts).

Regarding claim 18, the method as described in Claim 11, wherein a global fine tuning process is employed to automatically adjust "greediness" of computational algorithms used to perform said dynamic scheduling tasks (Col.13:53 – 63, see indicates efficiency and algorithm changed automatically).

Regarding claim 19, the method as described in Claim 18 wherein said computational algorithms are formulated with simple linear computational relationships which can be weighted by a reduction fraction (f) and a balance of said reduction fraction (1-f) based on recent and historical statistical data in said function call history model (Col.23: 1 – 20, for fraction see ratio).

Regarding claim 20, the method as described in Claim 1, further comprising the step of scheduling said function calls by performing a training mode, said training mode comprising the steps of:

by-passing said call history model based on benchmark data (Col.13:24 – 25, see segment table point and bypasses);

upon invocation of said each function call during an initial software program run-time, activating both said hard implementation and said soft implementation of said each function call (FIG.20, 2001 - 2006);

constructing a call history model with statistical data logged relating to said hard implementation and soft implementation of said each function call during said initial run-time (Col. 13: 55 – 60); and

upon subsequent invocations of said each function call, dynamically updating said call history model constructed during said initial run-time, wherein said call history

models is constructed on-the-fly during software program run-time (Col.13:53 – 63, for "on-the-fly" and dynamic, see while on line and algorithm changed automatically).

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claim 2 is rejected under 35 U.S.C. 103(a) as being unpatentable over Wood USPN 5,881,311 as applied in claim 1, in view of Hellerstrand et al. USPN 6,263,302.

Regarding claim 2, Wood further discloses the method as described in Claim 1, wherein said step of processing each function call identified within the software program into both a hard implementation and a soft implementation comprises the steps of:

inserting within said software program a first pair of code statements for identifying blocks of code corresponding to each function call, bounding each said block of code with a start statement in the front and an end statement at the end (Col.13:65 - 14:5, see insertion algorithm);

further inserting within each said block of code a second pair of code statements identifying sub-blocks of code within each of said block of code targeted to be executed in said reconfigurable logic partitions, each sub-block of code having an associated function performable within said reconfigurable logic partition (Col.13:57 - 60 see change algorithm);

transcribing said associated function of said each sub-block of code to generate each of said sets of configuration bits which, when loaded into the configuration memory of said reconfigurable computing system, causes one or more of the said reconfigurable logic partitions to perform said associated function of said each sub-block of code

(Col.16:48 – 55, for transcribing see "written" and for configuration bits see " CPUT determines whether the changed bits").

Wood doesn't explicitly disclose compiling a first code corresponding to said soft implementation and a second code corresponding to said hard implementation, wherein said first code is executed in said embedded processor, and assembling said first and second codes and configuration bits to form an executable code for said software program. However, Hellerstrand does disclose this feature in an analogous art (Col.29: 33 – 45, also see 30: 20 - 27). Therefore it would have obvious to one of ordinary skill in the art at the time the invention was made to combine Wood and Hellerstrand because, it enables implementations to produce target executable or host executable code.

Correspondence Information

8. Any inquires concerning this communication or earlier communications from the examiner should be directed to Chuck O. Kendall who may be reached via telephone at (703) 308-6608. The examiner can normally be reached Monday through Friday between 8:00 A.M. and 5:00 P.M. est.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam *can be* reached at *(703) 305-4552.*

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the Group receptionist whose telephone number is (703) 305-3900.

For facsimile (fax) send to central FAX number 703-872-9306 *and* 703-7467240 draft.

Chuck O. Kendall

TUAN DAM
SUPERVISORY PATENT EXAMINER